BOOST SOFTWARE BUSINESS PRODUCTS WITH HYBRID DEVELOPMENT

Miguel Oliveira,
Institute of Electronics and Informatics Engineering of Aveiro, University of Aveiro,
Campus Universitário de Santiago, 3810-193 Aveiro, Portugal
miguel@ua.pt
Rui Jesus,
GoFox – Information Technologies,
SanjoTec - Centro Empresarial e Tecnológico, 3700-121 São João Madeira, Portugal
rui.jesus@gofox.pt

ABSTRACT

This paper presents how new paradigms and methodologies for software development are changing rapidly in the last two years. In the current scenario where we live on, occurs a transition that, although slight, reflects the rapid manner in which the software production paradigms are reinvented due to the change of display devices and interaction with the end user. Studies indicate that in 2013 was the turn out of the internet access domain for mobile devices over the traditional desktop device, which is currently at around 60% mobile, against 40% desktop. This field will tend to grow in the coming years and it is expected that the use of internet for a desktop terminal tends to be less each day (comScore). In this context, the software industry has been re-invented and updated with respect to technologies that promote software and mobile applications, building products capable of responding to the user market. The development of software products, such as applications, must be put into production for different user environments, such as Web, iOS and Android in a way to enhance efficiency, optimization and productivity in the software development cycle (Langer, Arthur M.).

Keyword: Software, Hybrid, Native, Development, Mobile

INTRODUCTION

It is generally accepted that, despite the software development process have changed little or with small variations, the effort rate employed by organizations to develop new products tend to be difficult and costly due to the multiple options available with equipment and systems (Kohan, B.). Taking into account that manufacturers of components and programming technologies are sensitive to these issues, they have released a number of options that allow developers to develop hybrid applications rather than native applications (Reimler, S.). In short, software development organizations are increasingly opting for the hybrid paradigm, replacing the native, with the premise *develop once, deploy many times*.

The advantages of the hybrid development applications are innumerous: lower cost, reuse, efficiency with tests and error management, independence of the native technology, increase in productivity, savings in bandwidth consumption with servers, etc. It is, however, necessary that this new paradigmatic approach requires the implementation and use of processes and rules to ensure that developers concentrate their efforts in the software development cycle, logic applications and business rules.

THE NEW MODEL

Before the chance of using hybrid development tools, such as Cordova, Phonegap, Xamarin, organizations had to develop web, android, iOS and other platform versions of a software product. The costs are innumerous: different profile developers, aggressive headhunt recruiting, assertive management to accomplish same milestones for different platforms, platform specific tests and debugging, support, strong coordination, etc.

Besides the specific API platform development, the web model almost depends from server side scripting like PHP, Python, C#, etc., for data processing and content generation. Hybrid development tools enables new paradigm on development. They generate and deploy software for different platforms and display devices enabling software programmers to build applications for mobile devices using JavaScript, HTML5, and CSS3, instead of relying on platform-specific APIs. It enables wrapping up of CSS, HTML, and JavaScript code depending upon the platform of the device, extending features of HTML and JavaScript to work with the device. The resulting applications are hybrid, meaning that they are neither truly native mobile application, because layout is rendered via Web views, instead of the platform's native user interface, nor purely Web-based, because they are not just Web apps, but are packaged as apps to be published and have access to native device APIs (Fermoso, J.). Mixing native and hybrid code snippets has been possible from earlier 2009, although several issues were not accomplished in those days like plugins for mobile gadgets (accelerometer, battery camera, etc.). In result, server side scripting is becoming more often to use only for data exchange in JSON, JSONP or XML formats, liberating server from content and interface generation.

This new paradigm offers better options in terms of technology, support, adaptability, usability, accessibility, access to server management in the use of bandwidth, reuse, responsiveness, SEO (Google), among other things. In order to substantiate this approach, a set of methodological and procedural guidelines should be defined after the analysis and underpinning research, in the form of good practices guide, prepared for the automation of the process and its integration in the software development cycle. Subsequently, a set of components and applications should be developed with a view to reuse in different contexts, logic and business rules, leading to an efficient development framework.

REQUISITES FOR HYBRID DEVELOPMENT

An efficient development framework should be developed, in the basis of an abstract layer and/or model, with the premises of the following issues: adaptability, usability, accessibility, access to server data, reuse, responsiveness, SEO, easy plugin/API integration and platform deployment. It should be an easy-to-use framework, capable of a set of instructions, commands, templates that are gathered in scripts and style files (javascript, css). Reference and user manuals should be documented in order to provide help support for developers.

The strategy should endeavor to respond two major issues: By one hand, the need of simple and easy to use framework, sparing time to developers to dedicate their efforts on software development life cycle, rather than technologic issues. On the other hand, to promote reusability of software products in a world market that is, each day that passes, more mobile with different access devices. These issues must promote a better

sustainability of corporate organizations, i.e., reusability is what allows corporates to get more gains and income (Aked, M.). Following this context, a development framework, must be easy to use and integrate in software projects, that will deploy ready-to-market software/applications, independently from hardware, operating system platform and media interaction interface.

In order to lead this challenge to a final result, major problems are raised. On of the first is why should developers opt for hybrid development instead of native development, when there are many voices for and against hybrid solutions, such as platform manufacturers that have the monopoly of online app stores and strongly advice against hybrid solutions for publication?

After getting over previous questions, efforts should be addressed for what application builder/deployed should be chosen and why? There are many options in hybrid builders, so what should be the choice for developers? What are the mandatory requisites to make the choice? Functionality, fastness, number of platforms, support, robustness or other questions? Some research and query surveys should be made in order to congregate the best solution, always having in mind the premise that developers should focus their efforts on development cycle and also the user devices target.

Requisites to built through an abstract model should power flexibility in the making of applications. The abstraction layer must take in account several issues that enable wide end-user solutions:

a) An abstract interface layer, capable of implement different templates defined by users;

b) Abstract interface layer customized to different device interfaces (smartphone, tablet, desktop), ensuring responsiveness;

c) The abstract layer accomplishes standards for web, usability and accessibility;

d) The abstract layer must be able to be linked with datasets;

e) An abstract layer for datasets must be defined to allow developers to create their own dataset views or templates;

f) The dataset abstract layers needs to communicate with server by an exchange data type, probably json or jsonp ;

g) The framework ensures default asynchronous server access, parameterized, although, in special occasions, synchronous communications should be allowed;

h) The framework should induce developers to defined correctly metadata, in order to power their presence in web search engines (SEO);

i) The framework should be built-in with easy to use features form HTML5 APIs ;

j) A set of interface and dataset layers should be enabled in framework libraries. The framework libraries must me capable to include user defined templates;

k) An exemplary configuration file of the framework must be default defined, in order to make deployment easy. Instructions to install manufacturers sdk should be described;

l) All sources should be organized by types, in different files and folders;

m) Remote servers should only be used for data exchange, preferably in json or jsonp.

APPS NEARBY FUTURE

Since 2013, apps have been expanding for different devices, resulting in new app market opportunities for software companies. Now, not only available for mobile or desktop devices, but also for watches (Etherington, D.), TVs (Meyer, D.), Cars (Puri, N.), domotics, household appliances, projectors, boilers, etc.

If a software company aims to expand their software products for different devices and thus, maximize their market penetration, several issues are at stake: different technical profile collaborators (android, iOS, blackberry, device experts, etc.), diverse teams for support, maintenance and management, different software development approaches. Recently, Google confirmed that android framework will replaced in short term by the open source fork version of Oracle's Java (Protalin, E.ski). When a company promotes and sells consolidated products for a specific platform, a huge effort will be need to be made to migrate code for new technology. Besides, lately, technology manufactures have small time lapses to enable 3rd parties to do the job. In this context, using a hybrid software development methodology will shorten companies time window to the go-to-market milestone. Also, it powers your strategy and organization structure, in the way that software companies may invest in less, but highly skilled people for development, instead of having the need to hire multi developer profiles. Moreover, different industries segments are getting involved in groups to help and produce recommendations for software manufacturers like the Car Connectivity Consortium.

DESIRABLE RESULTS AND CONCLUSIONS

After these requirements are gathered in the form of requisite analysis and conception report, development should be started, analyzing the different technology issues: interface, data exchange, structure, configuration, deployment recommendations from consortiums and manufacturers. This will lead to efficient development framework, and should ensure the features described in previous section:

a) Support for interface in the form of methods and objects;

b) Support for data exchange in the form of methods and objects

c) Support for data interoperability in the form of methods and objects for CRUD (Create, Read, Update, Delete) functions;

d) Support for local and storage sessions, rather than cookies and sessions;

e) Support for HTML5 APIs;

f) Support for deployment in the chosen platforms;

The expected result should be a simple and easy to use framework that will allow developers to build applications on the fly.

In order to consolidate the efficient development framework, pilot applications will be developed using the proposed philosophy: client devices should do the job from user. Server will support the application, only through data exchange from database. Further, pilot applications release, metrics about the development time, material and human resources consumption are produced and compared with native development. Almost certainly, the results will prove the advantages of hybrid development recurring to a development framework. The main goal will be if the software development cycle is the major concern of developers, and if the redundancy of application logic and business rules are avoided due to the centralization of development.

The innovator characteristic of and hybrid development framework is to congregate, in a simple way for developers, hybrid deployment, interface design, data exchange to save server processing and connection bandwidth, SEO metadata to increase page ranking in search engines, responsiveness and use of recommended Consortium APIs. Base technologies are available in the market although are dispersed and not prepared to enable large scale application production.

The main idea came up through several needs discussed with GoFox, an SME partner of University of Aveiro, that has been hiring former students from short cycle higher education on development software, after they end the internship. GoFox, a dedicated company for SEO and app deployment, feels this gap in order to increase efficiency and productivity, avoid redundancy and distortion of the software development cycle and with main goal of focusing their collaborators on the proposed challenges, rather than technologies.

REFERENCES

comScore. 2014. *The U.S. Mobile App Report*

Langer, Arthur M. 2012. *Guide to Software Development: Designing and Managing the Life Cycle*, Springer, ISBN 978-1447122999

Kohan, B. 2015. *Native vs Hybrid / PhoneGap App Development Comparison: A comparison of native app development (iPhone: Objective-C / Swift, Android: Java) vs hybrid / PhoneGap app development (HTML5, CSS, JavaScript)*. http://www.comentum.com/phonegap-vs-native-app-development.html (15th January 2015)

Reimler, S. 2015, *Switching from native iOS to Ionic: Why Hybrid doesn't suck (anymore)*, https://www.airpair.com/javascript/posts/switching-from-ios-to-ionic (15th January 2015)

Fermoso, J. 2009. *PhoneGap Seeks to Bridge the Gap Between Mobile App Platforms*. http://gigaom.com/2009/04/05/phonegap-seeks-to-bridge-the-gap-between-mobile-app-platforms/ (15th January 2015)

Google. 2015. *Webmaster Guidelines*, https://support.google.com/webmasters/answer/35769?hl=en (15th January 2015)

Aked, M. 2003. *Risk reduction with the RUP phase plan*

Etherington, D. 2015, *Apple Boasts Over 3,500 Apple Watch Apps Already Available*, Techcrunch

Meyer, D. 2012, *LG follows Sony by using cloud to link Android and TV*, ZDNET

Puri, N., 2013, *MirrorLink: Your vehicle is now a smartphone app*, ZDNET

Protalinski, E. 2015, *Google confirms next Android version will use Oracle's open-source OpenJDK for Java APIs*, Venturebeat